Student Name: **Student id:** **Sect #:** **Ser#:**

University of Bahrain College of Information Technology

Department of Computer Science

ITCS242: ASSEMBLY LANGUAGE PROGRAMMING **Quiz #4**: Arithmetic Instructions

****

QUESTION ONE: Assume that *f1*, *f2*, and *f3* are predefined signed bytes. Write No more than 8 instructions to calculate the value of *f* as shown below. (Not allowed to change *f1*, *f2*, *and f3*). **Define *f* as needed.**

$f = ((f1 * f2) \ \% f3) - (f2 * f2)$

```
f  Sbyte ?

mov  AL , f1
imul  f2
movsx  AX , AL
idiv  f3
mov  AL , f2
Imul  f2
Sub  AX , AL
mov  f , sbyte ptr ax
```

QUESTION TWO: What would be in the AX register after executing the following code?
Your answer MUST be in HEXADECIMAL

```
MOV   AX, 7F3CH
MOV   BX, 3D90H
IMUL  BL
```

AX = C0 ***H***

Student Name: **Student id:** **Sect #:** **Serial:**

University of Bahrain College of Information Technology

Department of Computer Science

ITCS242: Assembly Language Programming **Quiz #1:** Data Representation & Architecture

****

SHOW DETAILED WORK on the sheet back PLEASE!!!

11) The computer components are connected using 3 types of busses, name any 2 of them:

control bus and address bus.

12) The logical address consists of the following two parts:

Segment and offset

13) The instruction operands may be located in many places, name any two:

main memory locations and output port

14) The addresses used by programmers are called Symbolic.

The addresses that travel on the address bus are called physical.

15) Using 8 bits show how the computer stores +93 0 1 0 1 1 1 0 1

and −93 1 1 0 1 1 0 1.

16) The largest signed decimal number that can be stored in **20** bits is $+2^{n-1}-1 = +2^{19}-1$.

The smallest signed decimal number that can be stored in **20** bits is $-2^{n-1} = -2^{19}$

17) The binary number **11001011** is equivalent to unsigned decimal value 171

and signed decimal value −43.

18) If a computer has **24** address lines and **32** data lines, the maximum size of directly addressable main memory is 4 Mbytes.

19) In real-address mode, the logical address **3CEB:5F77** is converted to the physical address

ph = seg *10 + offset = 3CEB0 + 5F77 = 42E27

20) Using **8** bits to store numbers show how the computer performs the operation $(43 - 87)_{10}$.

+43 : 0 0 1 0 1 0 1 1

-87 : 0 1 0 1 0 1 1 1 ⇒ 25 1 0 1 0 1 0 0 1

Student Name: **Student id:** **Sect #:** **Ser#:**

University of Bahrain College of Information Technology

Department of Computer ScienceType equation here.

ITCS242: ASSEMBLY LANGUAGE PROGRAMMING **Quiz #5**: Procedures

****

1) Write a procedure named "***funX***" that accepts a value X of type short and returns F calculated as follows:

$$F = \begin{cases} x / 10 & \text{if } x \geq 50 \\ x \,\%\, 10 & \text{otherwise} \end{cases}$$

Write a procedure in a form that allows using **invoke statement**.

```
funX  proc  x: sword ,
mov   Ax , x   ; copy Parameter
CMP   x , 50
JL    mod1
idiv  10
CBW
mov   f , AH
JmP   done
mod1: idiv  10
      CBW
      mov   f , AL
done:
```

2) *Given:*

x1	*sword*	*?*
f1	*sword*	*?*

Write ONE statement that applies the above developed procedure ***funX****, to calculate the value of f1 for x1*

```
invoke  funX , x1
```

Student Name: **Student id:** **Sect #:** **Serial:**

University of Bahrain College of Information Technology

Department of Computer Science

ITCS242: Assembly Language Programming **Quiz #2: Fundamentals**

****

- Answer the next 3 questions as needed

1) Give ONE instruction that uses indirect operand: MOV eax , [edi]

2) The range of values that can be stored using **sbyte** is -128 to +127

3) The directive that defines a constant **UU** equals to "ITCS242!" is

UU EQU "ITCS242!"

- Consider the following directive and answer the next 3 questions.

```
UU  sword  2A5CH,8FA4H,2 dup(-20,3 dup(-5, 2AH),7F4AH,9C4H)
```

4) The instruction that stores in **AX** register the number of bytes of array **UU** is

Mov Ax , sizeof UU

5) The instruction that swaps the first word of **UU** with **BX** is

Exchg Bx , Swordptr UU

6) The instruction that stores the value of **CX** in the **last** word of array UU in the above directive is

Movzx UU , CX

Mov UU [sizeof UU - 2] , CX

- Given an array: **ME SDWORD 240 dup (?);** Write Assembly code that swaps the words in each element of array **ME.**

```
.code
Mov ebx , 0 ; index
Mov ecx , 240
L1: exchg ME[ebx], ME[ebx + 4]   2 new locations
    INC ebx
    INC ebx
    INC ebx
    INC ebx
    loop L1
```

Mov ax , word ptr ME [ebx] (+4 -> or inc 4 times)

exchg ax , word ptr Me.[ebx + 2] (+4)

Mov word ptr [4 * ebx] , Ax

Student Name: **Student id:** **Sect#:** **#:**

University of Bahrain College of Information Technology

Department of Computer Science

ITCS242: Assembly Language Programming **Quiz #3: Input/Output Programming**

****

Write Assembly statements to perform the tasks in each of the following questions:

1) Define an array named **QUZ** consisting of 256 signed bytes.

```
QUZ   Sbyte   256 dup(?)
```

2) Fill the array **QUZ** by randomly generating 256 values in the range from **-64** to **+36** inclusively.

```
.code
CALL Randomize
Mov ecx , 0
Mov ebx , length of QUZ
L1: Mov eax , 100 ;  0 -> 100
CALL Random Range
Sub eax , 64  ; -46 -> 36
Mov QUZ [ecx] , eax
INC ecx
loop L1
```

3) Display in **HEXADECIMAL** all values of array **QUZ** as doublewords separated by spaces at the beginning of a new line.

```
CALL CRLF
Mov esi , QUZ
Mov ecx , length of QUZ
Mov ebx , Size of QUZ * 4
CALL DumpMem
```